ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
Εθνικόν και Καποδιστριακόν Πανεπιστήμιον Αθηνών
ΙΔΡΥΘΕΝ ΤΟ 1837

Course: Machine Learning

ASSIGNMENT: 1

FULL NAME / SERIAL NUMBER: GAVRIELATOS MARIOS / 7115152100023
GIANNAKI EVANGELIA / 7115152100025
SVOLOU STAVROULA / 7115152100038

Problem 1

We consider the generalized linear regression problem that is defined by the following model:

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_5 x^5 + \eta \qquad (1)$$

Where η corresponds to white Gaussian noise and the components of the weight vector assume the following values:

$$\theta_0 = 0.2\,,\ \theta_1 = -1\,,\ \theta_2 = 0.9\,,\ \theta_3 = 0.7\,,\ \theta_5 = -0.2 \qquad (2)$$

Initially, we observe that the coefficient of the 4th power, in our 5th degree polynomial, is equal to 0, in other words, we have $\theta_4 = 0$.

In aim to manage easier our model (1) and be able to apply the requested Regression methods, we will make some suitable conformations to gain the form of matrices production, as it seems below.

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_5 x^5 + \eta = \theta_0 + \theta^T \chi + \eta = [\theta^T, \theta_0] \begin{bmatrix} \chi \\ 1 \end{bmatrix} + \eta$$

or in short:

$$y = \theta^T \Phi + \eta$$

Where the vector θ now includes and the θ_0 value and Φ is the vector χ extended by 1.

In every experiment that we are going to conduct below, we will consider N equidistant points:

$x_1, x_2, \ldots, x_n$ in the interval $[0, 2]$

which will be used to create samples for our training set:

$$y_n = \theta_0 + \theta_1 x_n + \theta_2 x_n^2 + \theta_3 x_n^3 + \theta_5 x_n^5 + \eta_n, \quad n = 1, 2, \ldots, N \qquad (3)$$

where η_n are independent and identically distributed noise samples, that are derived from a Gaussian distribution with 0 mean and σ_η^2 variance.

1.1

For the first experiment we considered 20 equidistant points $x_1, x_2, \dots, x_{20}$ in the interval $[0, 2]$, σ_η^2 = 0.1 and the true generalized, 5th degree polynomial with the 4th power equal to zero, regression model. Then we define Φ and θ, as described above, and produced identically distributed noise samples, that are derived from a Gaussian distribution with 0 mean and σ_η^2 variance. In order to estimate the parameter vector, we have to minimize the cost function, which corresponds to the total squared-error loss. To achieve this, we have to calculate the derivative of the cost function with respect to θ and equate it to the 0 vector. The equation which is produced has the following form:

$$(\Phi^T \Phi)\hat{\theta}_{LS} = \Phi^T y$$

Consequently, the Least Square estimator of the parameter vector is calculated by:

$$\hat{\theta}_{LS} = (\Phi^T \Phi)^{-1} \Phi^T y$$

We applied the Least Squares method to estimate the parameter vector. An estimated parameter vector is the following:

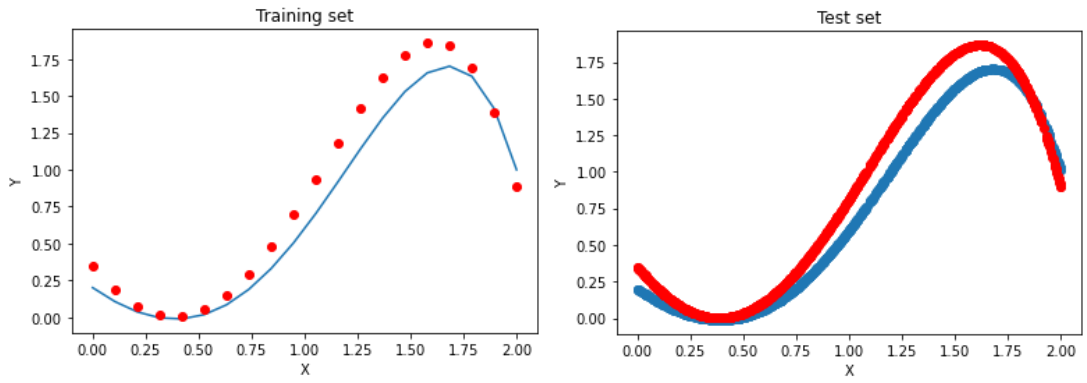
[0.421, -4.383, 13.835, -16.135, 8.8065, -1.791]

The estimated vector is not always the same because of the random noise.

The same holds for the Mean Square Errors of y over the training set and over the test set. We can now calculate the Mean Square Error (MSE) of y over the training set and over a test set comprising of 1000 points randomly selected in the interval [0,2].

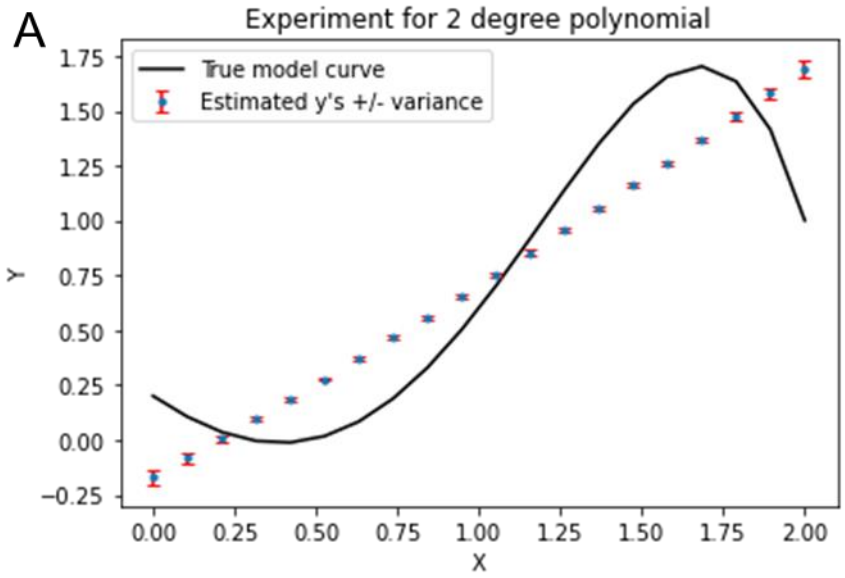
For the aforementioned parameter vector the MSE are 0.06743 for the training set and 0.12454 for the test set. The higher MSE for the test set was expected because the parameter vector was estimated using the training set.

The following plots show the True model (blue line/dots) against the predicted points (red dots) for the training set and the test set:

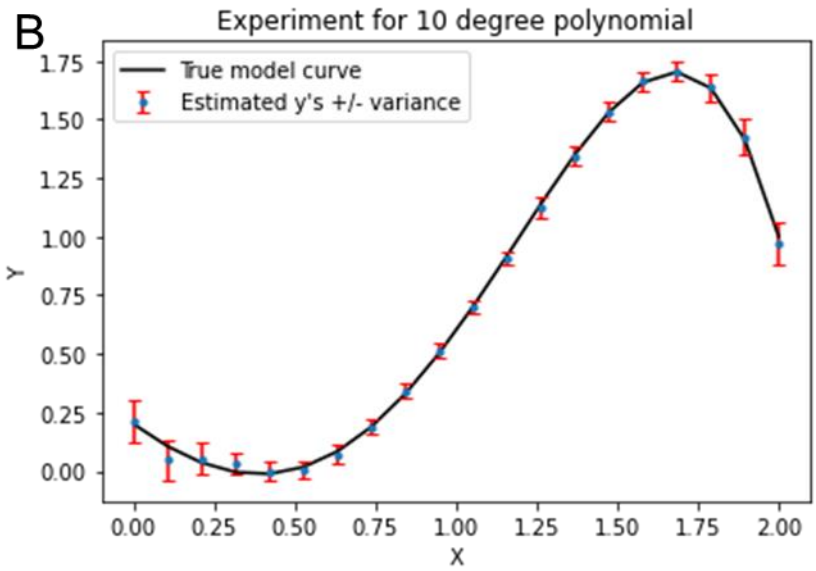


1.2

Using N = 20 and σ_η^2 = 0.1 we applied regression using the Least Squares method. We performed 100 experiments using two models of different polynomial degree. For the first experiment we used a 2nd degree polynomial model (lower degree than the true model) and for the second experiment we used a 10th degree polynomial model (larger degree than the true model). The curve of the true model (black line) is plotted against the mean and the variance of y over the 100 experiments (blue dots and red error bars):



Plot A indicates that the lower degree of polynomial, thus lower complexity, than the true model reduced the variance (smaller error bars) and increases the bias. This was expected as by definition bias error comes as a result of our tries to simplify the assumptions that are used in the true model, to make the approximation of the desired functions easier. By inserting the 2nd degree polynomial, while the true one has higher degree (5th). We know from the beginning that we are going to miss some principal relations between our regression simulations and the feature of our real model. Consequently, the problem of underfitting, i.e the impotence of the algorithm to identify the significant associations, was anticipated as it happens and with the existence of low variance. To fix the underfitting problem we must increase the variance and as a result the bias will decrease (bias-variance dilemma). This can be done by increasing the number of parameters in the model, the complexity of the model.



On the contrast, we observe in plot B of the 10th degree polynomial that we have low bias and high variance. By taking a higher degree polynomial than this of the true model, we are augmenting the possibilities to pay more attention in the training set than it is needed. This could be a problem as the algorithm could not correspond properly to a test set that has not seen again before and this could lead to an overfitting problem. For this reason, this model is unable to generalize well. An algorithm with high variance, as it happens with the 10th degree polynomial, is extremely sensitive in every fluctuation in the training set, even in the small ones. To avoid overfitting, we must increase the bias and as a result lower the variance (bias-variance dilemma). This is possible by reducing the complexity of the model by decreasing the number of parameters or using regularization and by increasing the size of the training set.

The bias-variance trade-off is a classical and important issue for all those who are dealing with the building of machine learning algorithms. By increasing bias, the variance decreases and when you increase variance the bias decreases. The dilemma lies in the effort to balance between the correct bias and variance values. Unfortunately, the boundaries of the trade-off are rigid and so it is not acceptable to make an algorithm flexible enough, in aim to fit properly in the training set, as the variance could be increased dramatically.

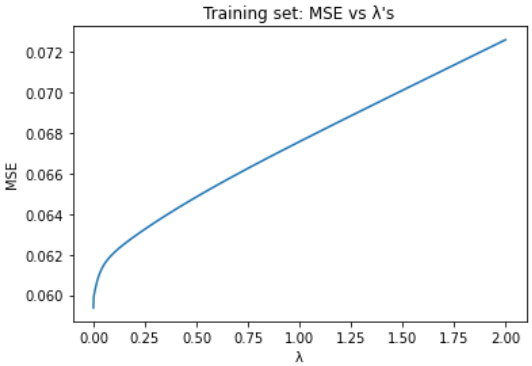
1.3

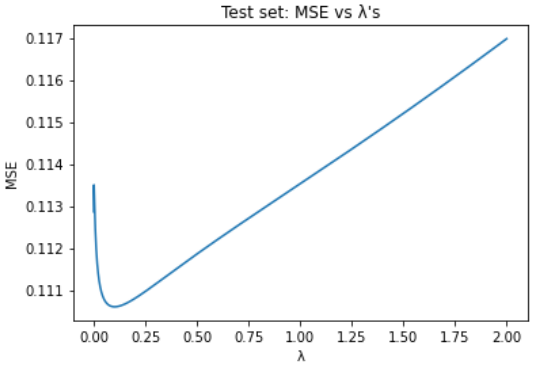
In order to implement the Ridge Regression method, we have to modify a little bit the Least Squares method algorithm. To be more specific, we have to add a constrain in the norm of the parameter vector, to limit the area in which we will be looking for the solution. This is happening by adding the λ‖θ‖^2 term in the LS loss function.

To achieve this, we have to calculate the derivative of the cost function with respect to θ and equate it to the 0 vector. Consequently, the Ridge Regression estimator of the parameter vector is calculated by:

$$\hat{\theta}_{RR} = (\Phi^T \Phi + \lambda I)^{-1} \Phi^T y$$

We repeated experiment 1.1. by implemented Ridge Regression method with various values of λ instead of using the Least Squares method. We tested different λ's from 0 to 2 with step 0.00001 in order to have high accuracy. For the training set, as shown in Plot A, the MSE was smallest for λ = 0, and for the test set the MSE was smallest (0.1106) for λ = 0.10109. The higher optimal λ value for the test set was expected because the model was trained on the training set. As the λ value increases so does the MSE value for both the training set and the test set.





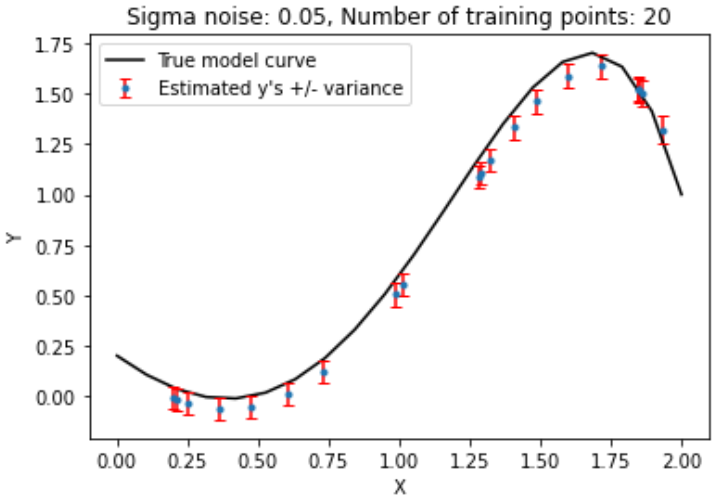
In general, we observe that we have an improvement of the Mean Square Error while λ is close to the value 0.1. Least Squares method is an unbiased estimator and this may lead to an overfitting problem. On contrast with Least Square, Ridge Regression introduces bias and, as we expected, it is able to to generalize better and score a smaller MSE value on the test set by reducing the overfitting (bias-variance trade off).

1.4

We implemented full Bayesian Inference in order to evaluate 20 randomly selected test set points belonging to the interval [0,2] and for two different values of σ_η^2 (0.05 and 0.15). We used as a mean of the Gaussian distribution the θ_0 of the true parameter vector ($\theta_0 = 0.2$) and covariance matrix $\Sigma_\theta = \sigma_\theta^2 I,\ \sigma_\theta^2 = 0.1$ as our prior knowledge for the unknown parameter vector.

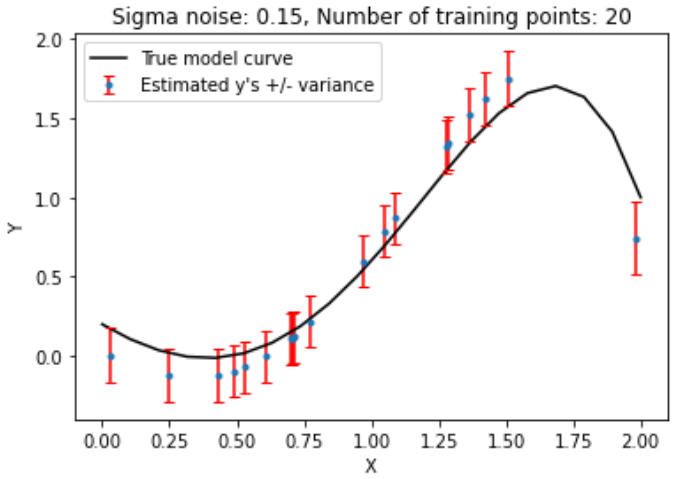
- For σ_η^2 = 0.05:

Using a small noise variance, we observe that the predictions are close to the true curve of the model. Moreover, the computed variance -uncertainty in my estimate for theta- is small (red error bars). Some estimates are outside of the true curve but we could improve our predictions by increasing the number of points in our training set.



- For σ_η^2 = 0.15:

Using a larger noise variance, we observe that our estimates do not fall on the true curve. The computed variance of our predictions is a lot bigger. The higher the noise variance of our original model the larger the error bars become.

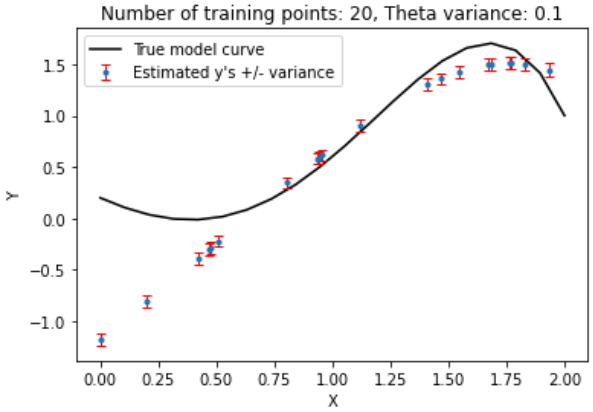


1.5

In this experiment, we repeater problem 1.4 but we used a non-true mean vector for G(θ): $\boldsymbol{\theta}_0 = [-10.54, 0.465, 0.0087, -0.093, -0.004]^T$ and σ_η^2 = 0.05. The experiment was performed 4 times by using two different values for σ_θ^2 (0.1 and 2) and two different values for *N* (20 and 500).

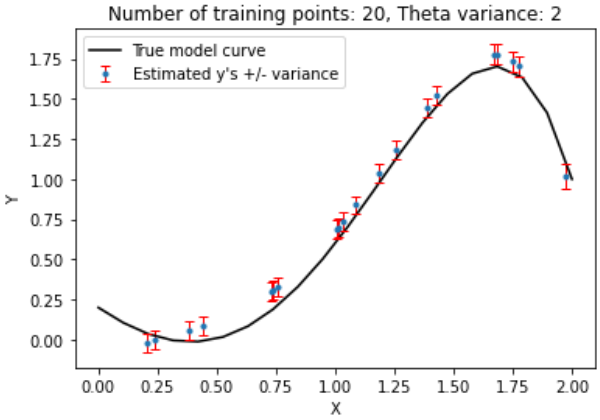
- For σ_θ^2 = 0.1 and *N* = 20:

Using a small prior variance, we indicate a small uncertainty, thus a high confidence about our mean vector. This, in combination with a small training set, results in a bad prediction.



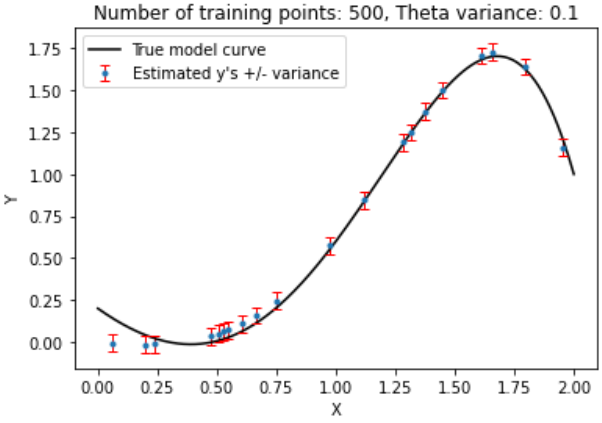
- For σ_θ^2 = 2 and *N* = 20:

Using a large prior variance, which indicates a big uncertainty about our prior knowledge, and a small training set, we achieve a better result compared to the first case. However, it is still not optimum.



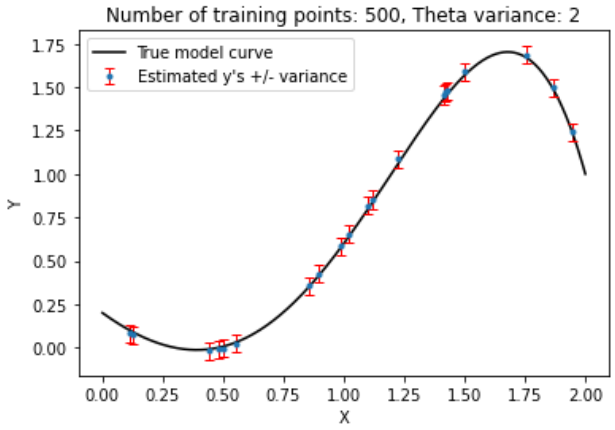
- For σ_θ^2 = 0.1 and *N* = 500

By increasing the size of the training set we are able to obtain a better result with lower bias.



- For σ_θ^2 = 2 and N = 500

By increasing the variance and by using a larger training set, we are able to obtain the optimum results.

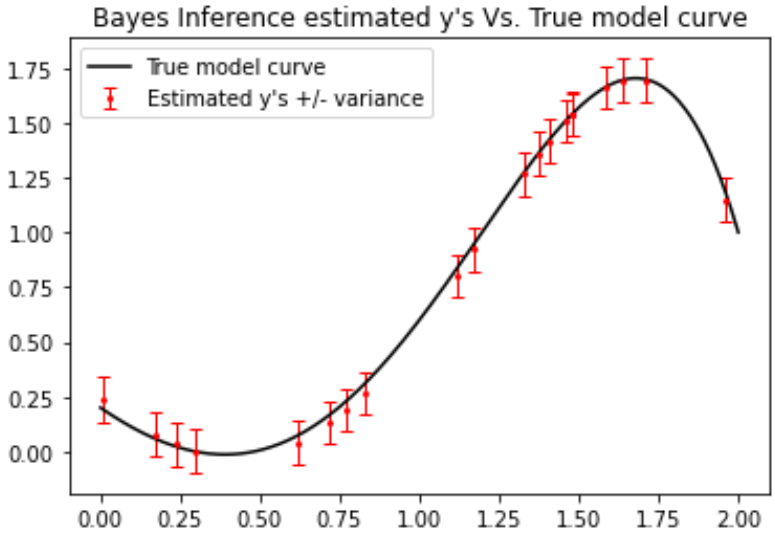


The experiments above indicate that by increasing the variance of the prior and by increasing the number of points in the training set, the bias is reduced and the predicted model approaches the true model.

By taking into account the results of the problems 1.4 and 1.5, we conclude that in order to use the Full Bayesian Inference we need a very large training set and a good prior knowledge about the model.

1.6

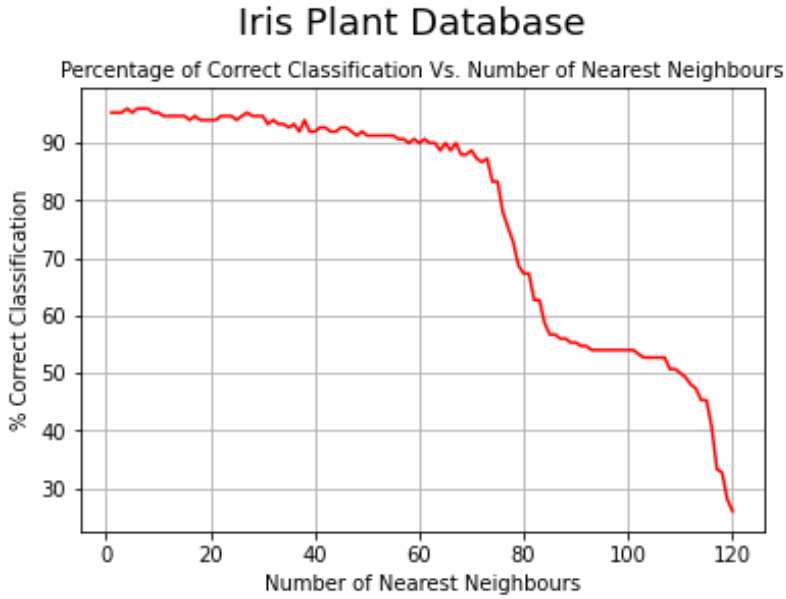
In the final experiment we implemented the Expectation-Maximization method in order to recover the variance of the Noise. We were able to estimate the noise variance in a high accuracy (True noise variance = 0.05, Estimated noise variance = 0.0486). The theta variance was also estimated (Theta variance = 2.602). Using these we implemented Bayes Inference to estimate the y's and their errors over a test set of 20 points randomly selected in the interval [0,2].



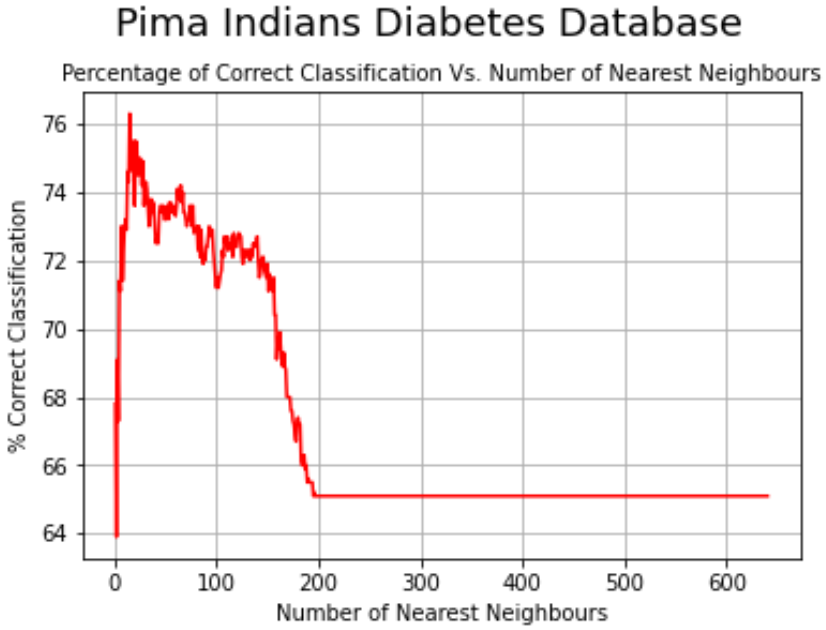
Problem 2

2.1

For each database, we implemented the k-NN classifier for all the possible values of k and performed cross-validation to estimate the percentage of correct classification for each k. In the case of Iris plant database, we performed 5-fold cross-validation for k=1 to k=120 (k_{max}=N-N/5, N=150). The optimal k for a specific run is k=9, with the percentage of correct classification equal to 98.0%.



In the case of Pima Indians diabetes database, we performed 6-fold cross-validation for k=1 to k=640 (k_{max}=N-N/6, N=768). The optimal k for a specific is k=15, with a percentage of correct classification equal to 76.3%.



2.2

a) We assume that the two classes follow normal distributions and that the covariance matrices are diagonal (components of the feature vectors are mutually statistically independent), with all diagonal elements equal (the variances of the features are equal). As we expected, this assumption achieves the highest scores in both AIC and BIC criterion for the two classes, meaning that the goodness of fit of this model, is not as good as that of the rest 3 models. For the two criteria, we used k=9 (parameters of the mean vector plus the variance).

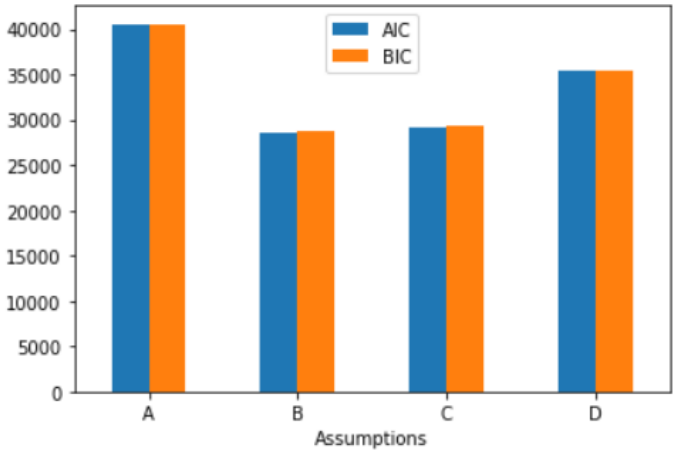
b) We assume that the two classes follow normal distributions, with non-diagonal covariance matrices. This model is better than the previous one, as it is able to capture more accurately the variance of each feature independently and how to features covariate. The analysis showed that this is the best model out of the 4 we tested, with the minimum AIC and BIC scores for both classes. For the two criteria, we used k=44 (parameters of the mean vector plus the parameters of the covariance matrix which is symmetric).

c) We assume that the two classes follow normal distributions, with the components of the feature vectors being mutually statistically independent. This assumption is better than the assumption in (a), as we calculate the variance of each feature independently rather than using the same variance for all features, however, it is not as good as the assumption in (b), because we predetermine that the feature vectors are mutually independent. Indeed, the analysis showed that this model is the second best. For the two criteria, we used k=16 (parameters of the mean vector plus the parameters the diagonal covariance matrix).

d) We assume that the components of the feature vectors are mutually, statistically independent and the marginal pdfs are computed using 1-D Parzen windows with gaussian kernels (h=sqrt(N)). This is the third best model out of the 4 in total, according to the AIC and BIC criteria for both classes. For the two criteria, we used k=1 (the h parameter). We did not consider the mean vector and the variance of the pdf as estimated parameters in the model because they are derived from the dataset.

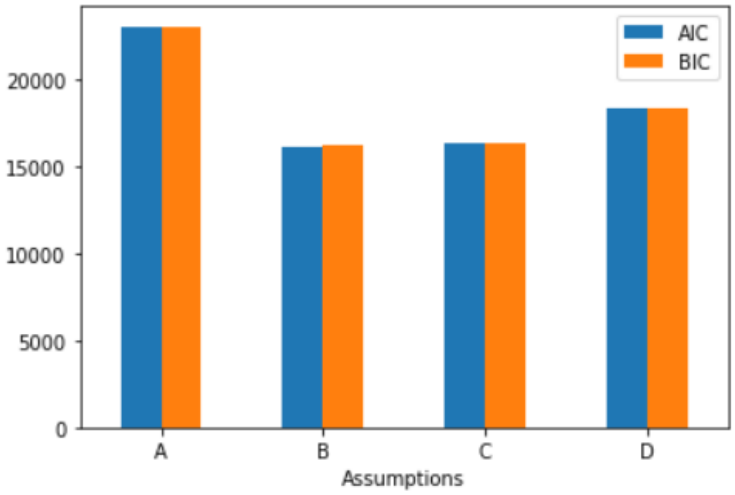
Class 0 (tested negative for diabetes)

```
Results for Class 0:
ASSUMPTION A: AIC = 40515.922320172955 and BIC = 40553.853793058755
ASSUMPTION B: AIC = 28576.82331525886 and BIC = 28762.266071589434
ASSUMPTION C: AIC = 29221.34727233397 and BIC = 29288.781001908723
ASSUMPTION D: AIC = 35440.99975100932 and BIC = 35445.21435910775
```



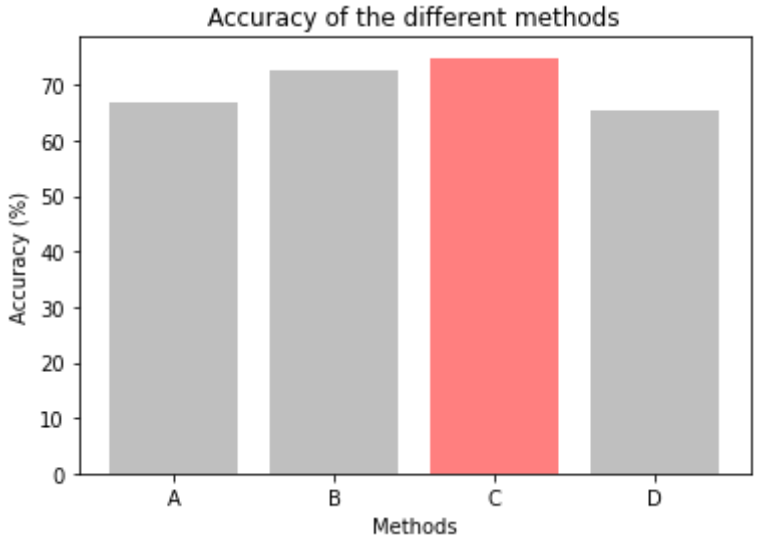
Class 1 (tested positive for diabetes)

```
Results for Class 1:
ASSUMPTION A: AIC = 23053.769172013224 and BIC = 23086.088054837823
ASSUMPTION B: AIC = 16111.863890855151 and BIC = 16269.867317997629
ASSUMPTION C: AIC = 16316.767199082524 and BIC = 16374.222990770697
ASSUMPTION D: AIC = 18416.273162215195 and BIC = 18419.864149195706
```



2.3

As the results of problem 2.2 denote, the methods B and C produce the best models. We also reached the same conclusion by calculating the accuracy of the different methods. The models A and D have the lower accuracies (65-67%) and the models B and C have the higher accuracies (72-74%). Taking into account the results of the problem 2.2 and 2.3 we conclude that more accurate estimates for the pdfs improve the classification accuracy as well.



The k-NN classifier can have similar scores to the Bayes classifiers and sometimes surpass them. The Bayes classifiers were computed much faster than the k-NN classifier.

2.4

The results show that only the class *Iris setosa* is linearly separable from the other classes when taking into account all the features of the dataset. The perceptron algorithm managed in 4 epochs to find a separating hyperplane:

$$1.3 \cdot x_1 + 4.1 \cdot x_2 - 5.2 \cdot x_3 - 2.2 \cdot x_4 + 1 = 0$$

On the other hand, the perceptron algorithm did not manage to find a separating hyperplane which will separate the *Iris versicolor* or the *Iris virginica* class from the other classes combined.